# Botnets and Network Analysis

Created by: Scott Kornblue

sakornbl@ucla.edu

With materials and sources originally developed by: Peter A. H. Peterson and Dr. Peter Reiher, UCLA {pahp, reiher}@cs.ucla.edu

Last updated: August 17, 2014

## Overview

The objective of this lab is to become familiar with basic network analysis utilities in addition to diagnosing and ultimately repairing abnormal network activities.
Concepts from previous lab activities will be utilized and expanded upon so you should have a solid grasp of all fundamental activities performed in previous labs.

After successful completion this lab, you will:

1. Understand the basics of network analysis.
2. Understand the basics of botnet operation and detection.
3. Understand how to diagnose network captures visually and interactively.
4. Gain familiarity with network analysis utilities.
5. Develop an understanding of techniques to reactively reply to a botnet attack.
6. Become familiar with techniques to help prevent new botnet attacks.

# Definitions

*What is a botnet? What is a bot? How are the two interconnected?*

The most simplistic way to define these terms would be to call a botnet a network of infected computers or "bots". "Bots" are most commonly instantiated by malware that allows an attacker to take control over an affected computer. Bots sneak onto a person's computer in many ways. Bots often spread themselves across the Internet by searching for vulnerable, unprotected computers to infect. When they find an exposed computer, they quickly infect the machine and then report back to their controller. Their goal is then to stay hidden until they are instructed to carry out a task.

In many real world situations bots have remained hidden for months, sometimes even years before exposing themselves by performing some type of malicious activity.
Networks of infected bot computers have been used to distribute spam email, host illicit web sites and services, propagate malware, and collaborate together to launch DDoS attacks against a specific target. DDoS (Distributed Denial of Service) attacks are one of the most common reasons why a network of bots could choose to remain hidden until the controlling element has achieved a satisfactory amount of infected bots to launch the attack.

*How do you know if a computer is part of a botnet?*

Botnets can be difficult to detect. Sluggish system performance and high utilization of resources are a common sign of malicious activity but not necessarily botnet activity.
In some situations, there may be no outward signs of botnet activity from a localized standpoint. Criminals try to hide their malware in an effort to infect as many computers as possible.
Surefire ways of determining botnet activity would be via an analysis of network interface traffic patterns, however, this is not easily accessible to the average user.
Internet service providers are beginning to take a proactive approach by issuing notices to customers when botnet traffic has been detected from their devices.
In addition, many private networks are beginning to implement specific network intrusion and network state procedures to identify patterns that do not meet baseline figures.
The element to take away from a botnet oriented malware vector when compared to conventional infections would be the network link or interface being just as much of a target as the computer itself.

*How do you prevent botnets?*

There are several measures that users and administrators alike can take to prevent botnet malware infection. Since bot infections usually spread via malware, many of these measures actually focus on preventing malware infections. Recommended practices for botnet prevention include:

Network baselining: Network performance and activity should be monitored so that irregular network behavior is apparent.

Software patches: All software should be kept up-to-date with security patches.

Vigilance: Users should be trained to refrain from activity that puts them at risk of bot infections or other malware. This includes opening emails or messages, downloading attachments, or clicking links from untrusted or unfamiliar sources.

Anti-Botnet tools: Anti-botnet tools provide botnet detection to augment preventative efforts by finding and blocking bot malware before infection occurs.
Most applications also offer features such as scanning for bot infections and botnet removal as well. Firewalls and antivirus software typically include basic tools for botnet detection, prevention, and removal. Tools like Network Intrusion Detection Systems (NIDS), rootkit detection packages, network sniffers, and specialized anti-bot programs can be used to provide more sophisticated botnet detection/prevention/removal.

*How do you react to botnet detection?*

Botnet detection is useless without having botnet removal capabilities.
Once a bot has been detected on a computer it should be removed as quickly as possible using security software with botnet removal functionality.
Once the process of botnet removal is complete, it is important to remain proactive in botnet detection and prevention efforts.

Botnet removal can go beyond simply removing bot malware from an infected machine.
On a larger scale, botnet removal often requires shutting down the controlling server that is used to control the botnet. This is typically done when an organization is looking to shut down an entire botnet rather than treat bot infections.
The FBI's campaign against the Coreflood botnet is a good example of large-scale botnet removal. (As part of your CS236 assigned reading you previously read about Coreflood here)
Another good read on this type of reactive activity can be located here

The main idea is while botnet activity can be cleansed from a local environment, the problem still exists elsewhere. The typical scenario is that you will not be in a position to do any more than just scrub your local environment. However, administration within a large scale local network could result in a bot controller node implemented within your own local network.
Understanding the fundamentals on how botnets operate, how they gain access into a compromised system and how procedures are implemented to deal with bot infections composes a valuable arsenal of knowledge to any network administrator.

More real world examples:

In addition to the Coreflood example, another well documented botnet attack, detection, and resolution occurred in 2007 within the national network infrastructure of Estonia.
A synchronized flood of botnet traffic attacked Estonian networks until local infrastructure was effectively disabled.
Rather than removing the bot controller and disabling the botnet from the inside out, the strategy of utilizing sophisticated firewall filtering was used to slowly but surely filter out botnet traffic from Estonian networks.
A brief but detailed analysis on this type of botnet attack and resolution can be read here
In addition a brief video can be viewed here

The following exercise that you will begin will provide a simplified viewpoint that incorporates many of the elements detailed above, albeit on a much smaller scale.
The same fundamentals that you will use in this lab correlate the very same approaches and processes taken by worldwide network security analysts and experts.

# SOFTWARE TOOLS

Before delving into this experiment we will first go over some helpful software applications that will aid you in this activity.
We will build upon existing applications you have already become acquainted with in previous exercises while also introducing some new ones.

### ETTERCAP

We revisit ettercap, the application we used to perform man-in-the-middle attacks in a previous lab. The same fundamental spoofing techniques apply, this time for good, not malice!
A network analyst will find ettercap to be a very beneficial tool to monitor network activity and inspect data frames that traverse the network.
Ettercap provides ARP spoofing techniques to change the analyst's network perspective without physically moving to a new location.
This is particularly useful for situations where attack vectors only target certain nodes or interfaces.

A refresher on ettercap utilization:

1. Start Ettercap: `sudo ettercap -C –i ethX --` this starts Ettercap with the basic ncurses terminal interface over X interface.
2. Choose "Unified Sniffing" from the **Sniff** menu.
3. find the hosts on the network **(Hosts** menu)
4. select targets (Add hosts to a target by highlighting them and pressing 1 in the Hosts list.)
5. spoof their ARP tables (Under **MITM** menu)
6. and begin sniffing the network. (Under **Start** menu)

### TCPDUMP

Being able to capture network traffic and store it in a file for later analysis is a critical tool for any network analyst.
You are free to use any PCAP compliant network capture application; however, tcpdump is loaded and available within this lab.
We will use the same procedures regarding tcpdump that you learned in previous exercises.

A quick refresher on tcpdump utilization:

```
sudo tcpdump –I ethX –s 1500 –X
```

This command invokes `tcpdump` listening on 4thernet adapter X, with a snaplength of 1500 bytes; using a smaller snaplength will not always show all the packet data (especially for large packets).
The option –X decodes the data in hex and ASCII, similar to the display of hex editors like `hexdump`.

`tcpdump` will stream all decoded information to the terminal, too quickly to read.
You can use Control-C to kill `tcpdump` after enough time has passed and use your terminal's scrollbars to look at the output, use `less`, or use a terminal manager like `screen` for its scrollback features.

You can also dump the capture to a special binary capture format with the `-w` switch:

```
tcpdump -I ethX -s0 -w output.pcap
```

This writes your network capture to file using the PCAP standard.
For this lab in particular, obtaining PCAP standard files for later analysis is a critical factor for successful completion.

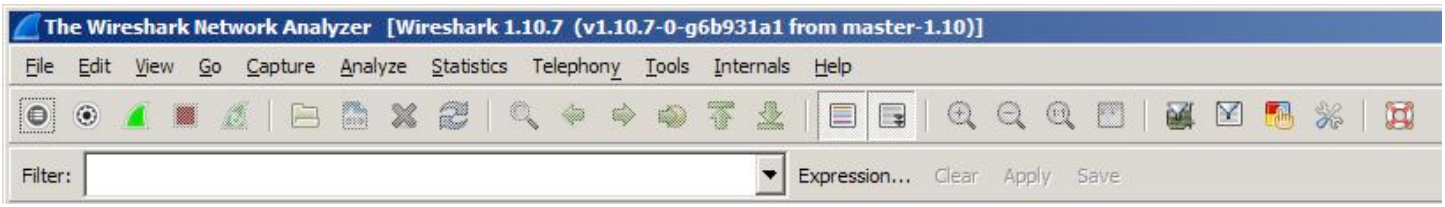
## WIRESHARK

Now a demonstration of what you can do with those captured PCAP files.
Wireshark is known as the "swiss army knife" of network capture and analysis applications. Among the many uses Wireshark finds in industry includes analyzing network problems, detecting network intrusions, network misuse, monitoring usage and gathering network statistics. You are free to use any compliant PCAP analysis application, however, we will demonstrate the usage of Wireshark in this lab and this would be recommended if you have no prior experience.

Unlike other utilities you have been exposed to, Wireshark is an application that needs to be installed on your own personal computer to provide a full featured graphical interface to aid in network analysis.

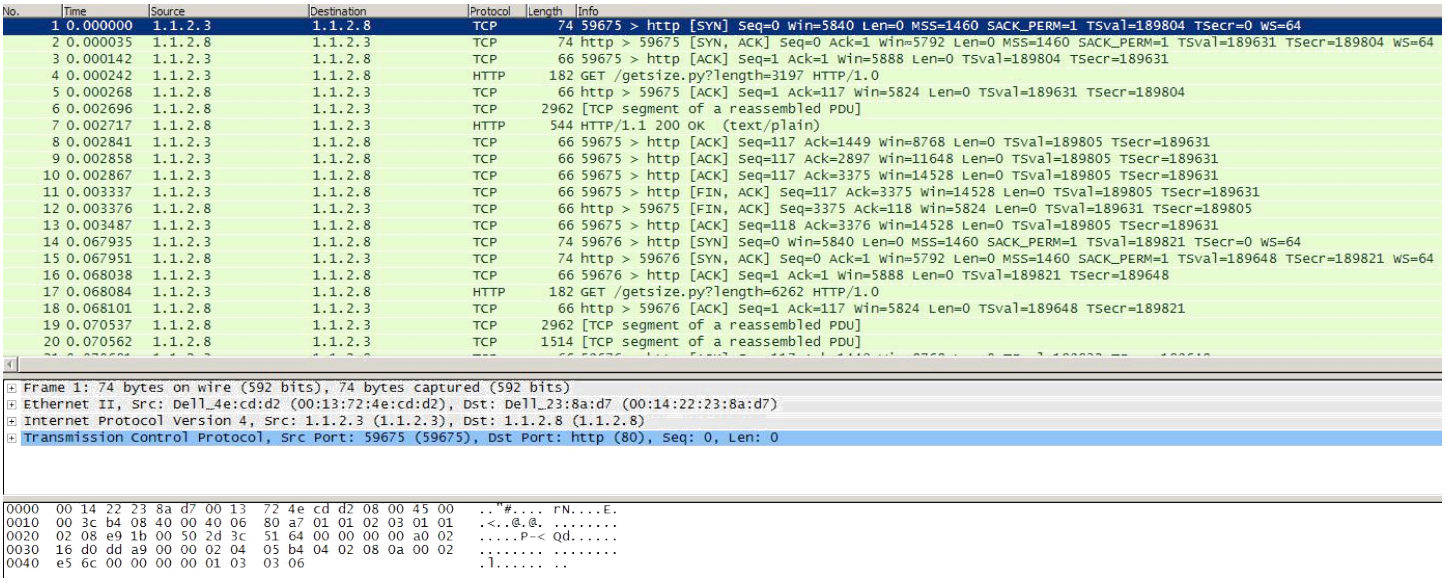
Wireshark is a free and open source PCAP analyzer that is available for download directly from the Wireshark Foundation

Once Wireshark is installed on your computer you can open any PCAP compliant network capture you create within the Deter testbed.
(Recall the Deter instructions on how to transfer files from Deter to your local computer. SCP is the easiest way. A recommended GUI application for Windows users is WinSCP)



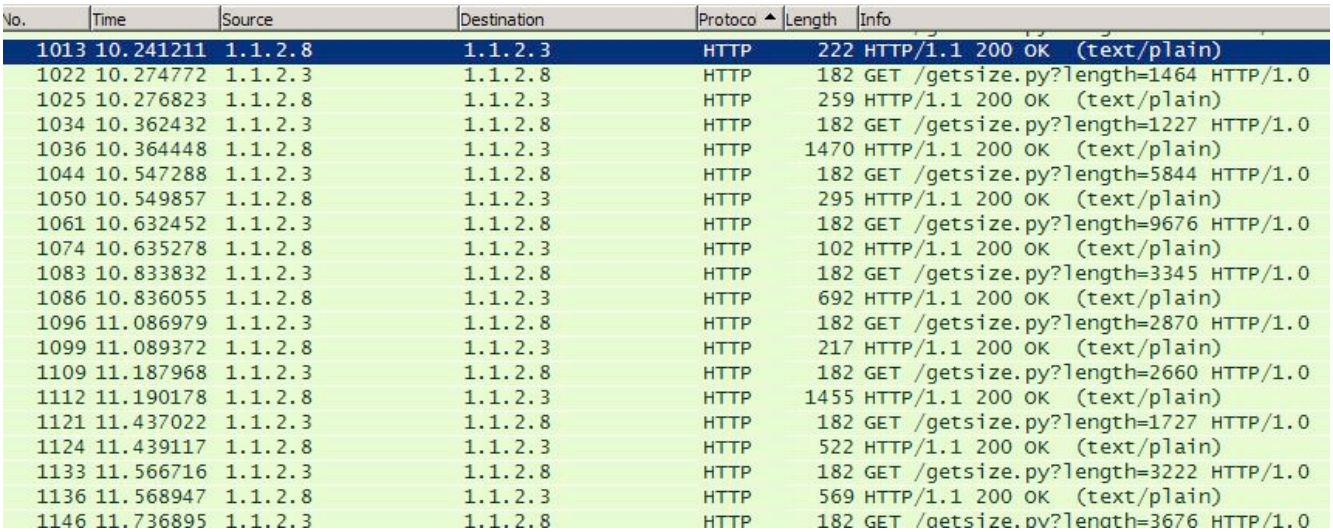

Wireshark will allow you to sort, filter, and manipulate PCAP files in graphical fashion using this graphical interface.

You will want to open the File menu and Open a stored PCAP file.

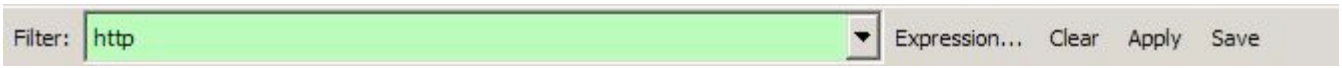



This is a sample depiction of what you will observe after loading a PCAP file. By default frames are sorted by time in sequential order. Underneath the sequence of frames a windowed dialog will display the frame contents for manual analysis on a frame-by-frame basis.

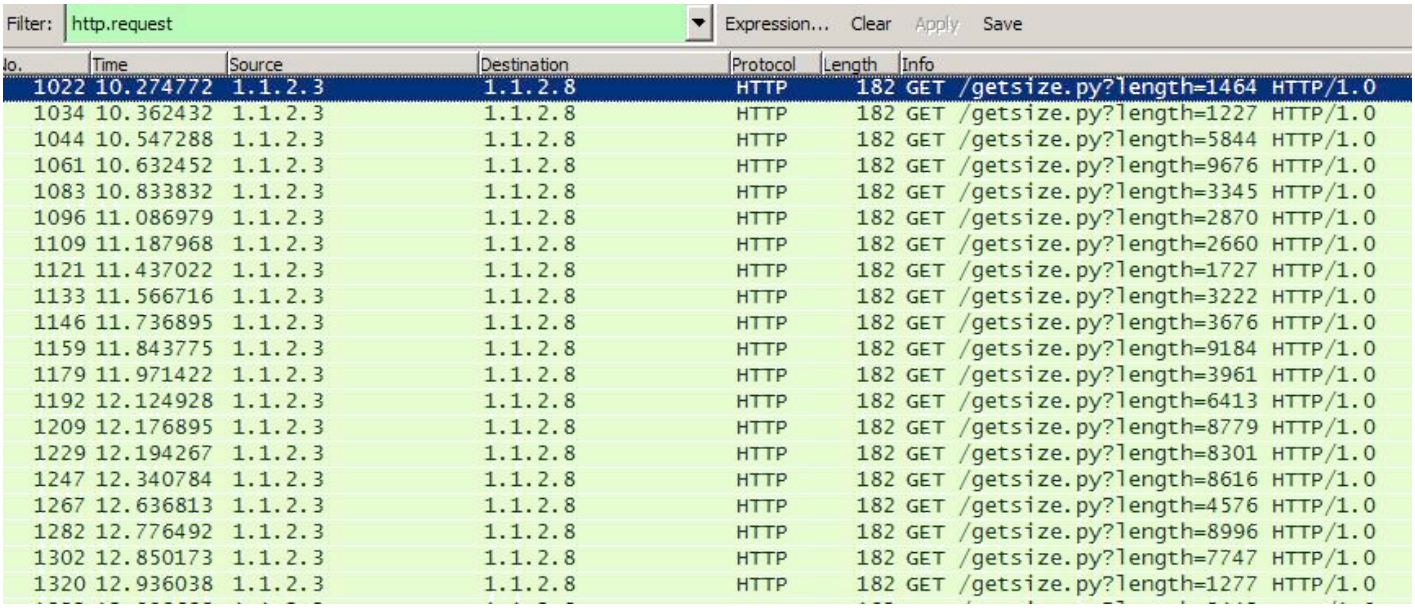
Frames can also be sorted by any constraint listed in the heading. Clicking Protocol will sort all frames by protocol as shown here:



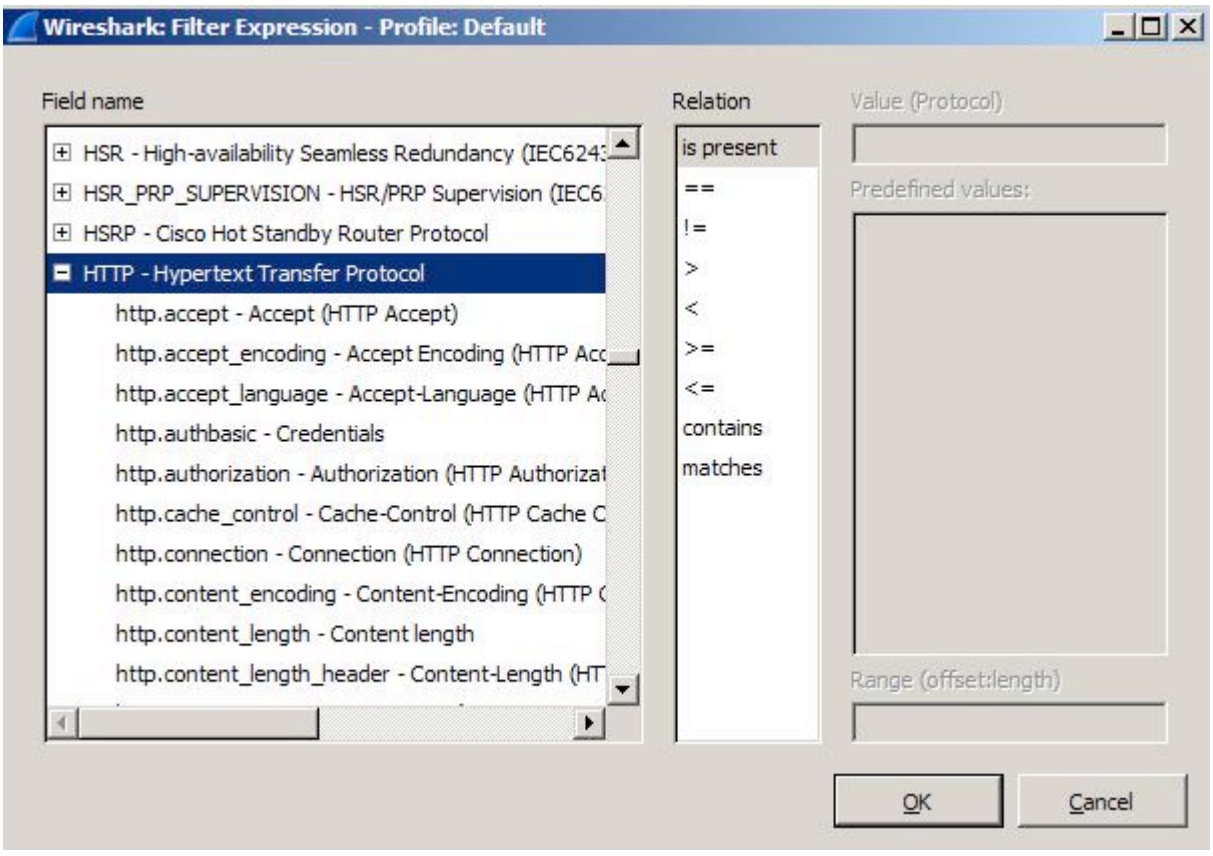

The same result can also be achieved using the Filter toolbar by searching for HTTP

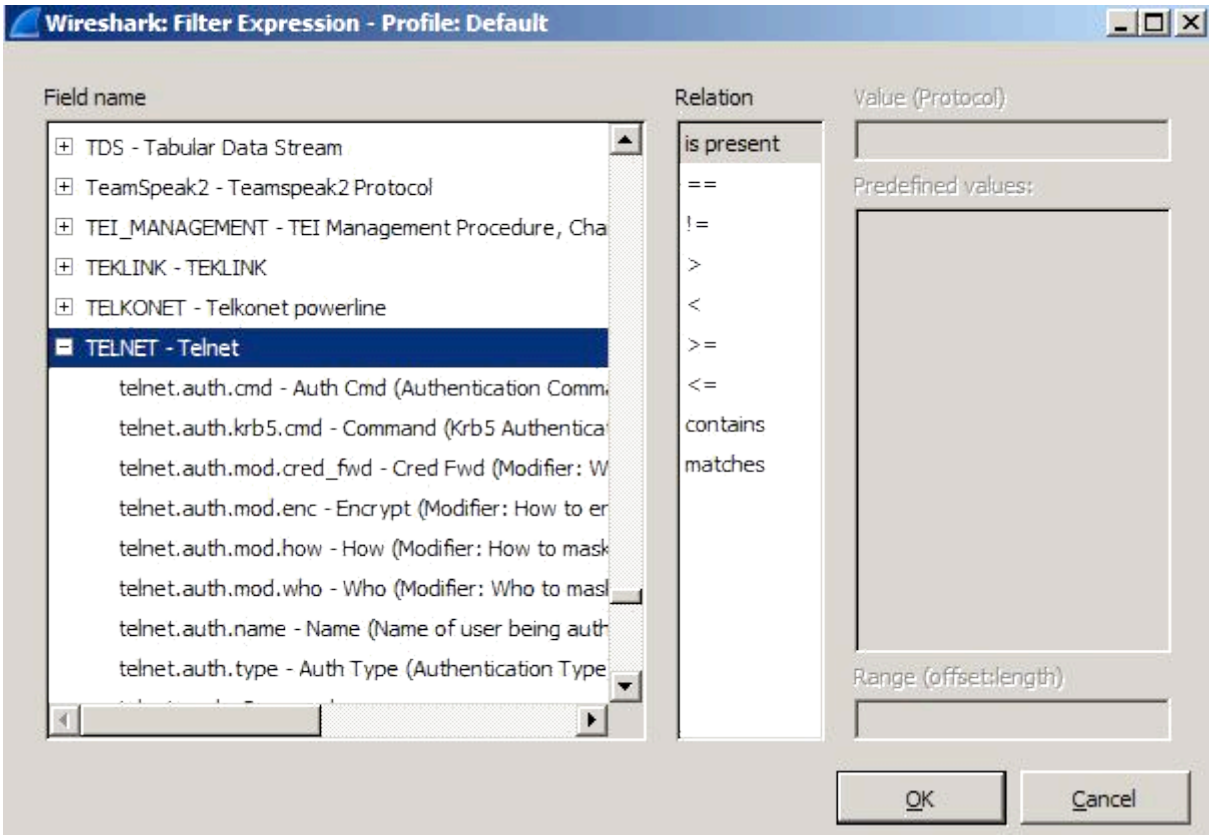



The Filter toolbar allows more sophisticated filtering to be performed
For example, to see only HTTP GET request frames the Wireshark filter http.request can be invoked to produce the following output




Clicking the Expression icon allows you to craft custom filters or pick ones from the built in Wireshark library.






The built in database includes many filters you could find useful for common protocols such as HTTP and TELNET
It is best to try things out and experiment on your own. A sample PCAP file is available here

**IPTRAF**

IPTraf is a basic network monitoring utility for IP networks. It intercepts packets on the network and gives out various pieces of information about the current IP traffic over it.
Similar to ettercap, iptraf is an ncurses-based IP LAN monitor that generates various network statistics including TCP info, UDP counts, ICMP and OSPF information, Ethernet load info, node stats, IP checksum errors, and others.

The basic invocation process for IPTraf is just executing `sudo iptraf`

Without any parameters given IPTraf will execute in an ncurses GUI.
It would be best to explore this GUI and try many functions out.
By default IPTraf will display a real time view of network interface traffic.

To specify a specific interface

invoke: `sudo iptraf -i ethX`

IPTraf offers a very useful option to monitor all interfaces simultaneously. This is especially useful for multi-interface nodes such as a router.
You can invoke IPTraf over multiple interfaces by

executing: `iptraf -i all`

IPTraf also allows you to filter the view to only show certain protocols during

viewing `iptraf -s ethX` will only monitor TCP and UDP traffic for instance

**TOP**

Top provides an ongoing look at processor activity in real time. It displays a listing of the most CPU-intensive tasks on the system, and can provide an interactive interface for manipulating processes. It can sort the tasks by CPU usage, memory usage and runtime. Most features can either be selected by an interactive command or by specifying the feature in the personal or system-wide configuration file.

Executing top is as simple as invoking `top` from your command line interface.
Once top is running further sorting and analysis can be performed by invoking the F key.

To change the delay between updates you can invoke

top by: `top -d XXXX`

To monitor only specific process names you can invoke top

by: `top -p XXXX`

**IFTOP**

While top displays real time CPU utilization, iftop displays real time network utilization.

Iftop can be invoked to a specific interface by

```
sudo iftop -i ethX
```

NOTE you must utilize sudo or be logged in as a root account for iftop to function.

While the iftop interface is active you can specify more options:

```
S - display source port
D - display destination port
n - show IP instead of host name
1/2/3 - sort by the specified column
< - sort by source name
> - sort by dest name
P - pause display ( or else it will be often updated to show the current status )
j/k - scroll display
? - for help
```

**NMON**

Some users will prefer to use nmon as a full featured combination of top and iftop. The nmon tool is helpful in presenting all the important performance tuning information on one screen and dynamically updating it. This efficient tool works on any command line or telnet session. In addition, it does not consume as many CPU cycles as older conventional utilities like top and iftop.

Nmon is a fully graphical interface that can be invoked using `sudo nmon` for full access.

Some helpful shortcuts once nmon is active:

```
q - To stop and exit nmon.
h - To see quick help (hint) screen and press h again to remove the hints.
Use the following command to turn on or off stats:
c - See cpu stats.
m - See memory stats.
d - See disk stats.
k - See kernel stats. n
- See network stats. N
- See NFS stats.
j - See file system stats.
t - See top process.
V - See virtual memory stats.
. - See only busy disks/procs.
v - Verbose mode (display as Ok/warning/danger on screen).
```

In addition, nmon could be employed non-interactively using the `nmon –f` invocation to run processes in the background and write to file.

For example the command: `nmon -f -t -r output.file -s 5 -c 200`

```
f – Save the output to file
r – Include CPU utilization
t – Include top processes
s – Seconds between refreshing the screen
c – Number of refreshes
```

As nmon operates in the background other processes can be executed and the resulting output file can be analyzed later within the nmon application or exported to read in any CSV compliant spreadsheet application.

As always, with all of these applications feel free to consult the man pages, research on the internet, or even locate and install any applications that you feel more comfortable using. Utilizing the lab within the Deter testbed lets you experiment without risk of screwing anything up. Feel free to adjust or otherwise modify the traditional usage of these applications in any way you see fit. All of these applications are very flexible and you as the network analyst have full latitude in how you intend to utilize them to meet your required objectives.

# INTRODUCTION

A local small business has received reports from their Internet Service Provider that unusual traffic patterns are being detected from IP addresses belonging to them.
You have been called in to investigate this unusual network activity. This business has several employee terminals and a local server used to provide web services to employees.

You have connected your own personal computer to investigate, accessible at:
`student.YourEXPname.UCLAClass.deterlab.net`

This unusual traffic occurs only during certain times of each working day. You have chosen to investigate on site during normal business hours in a time period where this unusual traffic is occurring. Be sure to make notes and keep track of what you have done and what reactions you observe in response to any actions performed.

The boss is infuriated that company productivity has been severely bottlenecked and is reliant on your abilities to restore order.
The better the solution that you propose, the more you will be paid.
Make sure you are clear and descriptive in your findings and decisive in your proposals.

## SETUP

1. If you don't have an account, follow the instructions in the introduction to DETER document.
2. Log into DETER.
3. Create an instance of this exercise by following the instructions here, using `/share/education/BOTNET_UCLA/botnet-4.ns` as your NS File.
4. In the "Idle-Swap" field, enter "1". This tells DETER to swap your experiment out if it is idle for more than one hour.
5. In the "Max. Duration" field, enter "6". This tells DETER to swap the experiment out after six hours.
6. Swap in your new lab.
7. After the experiment has finished swapping in, log in to the node via ssh.

It is **very possible,** almost definite that actions you take in the course of this lab could kill important processes, erase files, or generally cause other havoc in the system.
If you think that you have "broken" your lab nodes, you can always reboot the nodes, or if things are *really* screwed up, swap out the experiment and swap it back in to start fresh.
As always, anything you save in your group directory on DETER will stay there -- make sure you save any important work somewhere safe.

# TASKS

## 1. CAPTURE

Log in to your terminal and utilize as many network traffic diagnostic utilities as you see fit.
This procedure is very open-ended and as the investigator, you have free reign on how you wish to analyze.

HINT: Your knowledge of Ettercap and Tcpdump will be very handy.
**Do not scan any interface starting with 192.168.*.* -- that is DETER's control network!**

Capture at least 5 minutes of network activity in a way that provides an adequate viewpoint of the entire local network.

Briefly summarize how you captured network data, why you performed the actions you did and what (if any) alternative methods you also considered.

Please provide a network capture in PCAP format for submission.

## 2. ANALYZE

Delve into your network traffic capture and perform analysis.

Briefly summarize the type of traffic patterns that you observed within this network.

What kind of traffic observed would you categorize as unusual or malicious?
Be specific and provide examples.

How do you believe the local network fell victim to this unusual behavior?
Note any specific vulnerabilities you encountered during your investigation.

## 3. RESPOND

After reading over your recommendations, the boss has given you full clearance to rectify the problems found within the local business network.
You have been given full latitude in how you wish to solve the issue. Just make sure you do not hinder network functionality from working properly.

What are some methods that can be deployed to remedy the situation?

Is it within your abilities to fix the local
network? If not, why not?
If so, write a step by step analysis on what you did to restore order to the local network.

The boss has also asked you to analyze the overall security of local systems and system policies within the network.
What are some more intricate overarching security policies that you would recommend be deployed? Please suggest both technical and non-technical security improvements that you believe would benefit the network environment.

Were you able to locate any possible evidence of the origin of the abnormal network behavior? If so please document what you found and how you located it.
Provide rationale why you believe the evidence you located is malicious in nature.

After performing your repairs (if possible) perform the Capture process again and compare the traffic patterns Perform another PCAP capture and submit

### 4. EXTRA CREDIT

You have a very good hunch that the botnet operators did not safeguard their own controller very well.

What kind of evidence can you locate to support this?
Can you gain access to the controller? Why or why not?
Is it possible to completely repair the network without even needing to access or make any modification to a computer on the local network?
If so, please outline how this can be achieved.
Finally, can you exploit any found vulnerabilities to shut the botnet controller down entirely?

# Submission Instructions

Create a single .pdf, .doc, or .txt with all your text answers, screenshots, and anything else you think should be included etc., and submit it to your instructor. Make sure to double check that you have answered all the questions.
In addition, submit your before and after (if you have one) PCAP compliant network capture file (from Tcpdump or similar)
Please name your files Lastname_First_Before.pcap and Lastname_First_After.pcap